Grafika Komputerowa i Projektowanie
semestr 2, ćwiczenie 1

Moduł turtle - rysowanie fraktali

Moduł ***turtle***, będący częścią biblioteki standardowej języka Python, wykorzystywany jest najczęściej w celach edukacyjnych. Umożliwia on tworzenie prostych, dwuwymiarowych grafik poprzez sterowanie obiektem rysującym na ekranie, określanym żółwiem lub robotem. Obiekt ten najczęściej porusza się do przodu i do tyłu (rysując za sobą linie) oraz wykonuje obroty (zmienia kierunek, w którym się przemieszcza). Taką formę tworzenia grafiki przyjęło nazywać się grafiką żółwia a sam pomysł modułu ***turtle*** języka Python wywodzi się z języka Logo, stworzonego w 1966 przez Wally'ego Feurziga i Seymoura Paperta.

Moduł ***turtle*** do programu w języku Python dołączamy poprzez poniższą instrukcję:
`import turtle`
Z kolei obiekt żółwia w programie tworzymy wywołując inicjalizator (inaczej konstruktor) klasy `Turtle`:
`t = turtle.Turtle()`

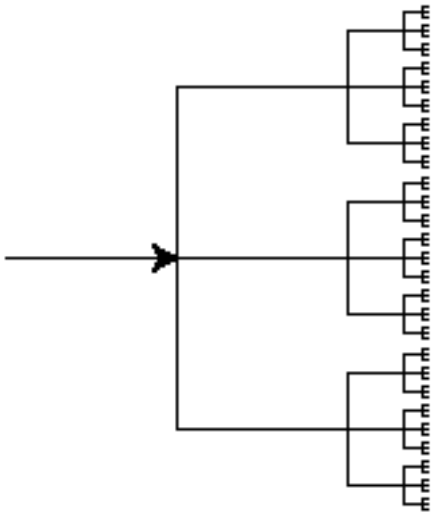
Sterowanie odbywa się poprzez wywoływanie odpowiednich metod na stworzonym w ten sposób obiekcie żółdwia. Podstawowe operacje to:
`t.forward(10)` – przesuń żółwia 10 pikseli do przodu
`t.backward(30)` – przesuń żółwia o 30 pikseli do tyłu
`t.left(45)` – obróć żółwia o 45 stopni w lewo
`t.right(60)` – obróć żółwia o 60 stopni w prawo

I tak narysowanie kwadratu o boku 10 pikseli to ciąg instrukcji:

```
t.forward(10)
t.left(90)
t.forward(10)
t.left(90)
t.forward(10)
t.left(90)
t.forward(10)
t.left(90)
```

lub zapisując to samo krócej - w postaci pętli:

```
for i in range(4):
    t.left(90)
    t.forward(10)
```



Rysunek 1: Prosty fraktal

Jeśli do tych prostych instrukcji rysujących doda się możliwość rekurencyjnego wywoływania funkcji, to można z wykorzystaniem tego modułu tworzyć fraktale (najczęściej obiekty samo-podobne). Jak więc przykładowo narysować prosty fraktal przedstawiony na rys. 1? Można to zrobić na wiele sposobów. Poniżej przedstawiono ogólny schemat postępowanie w tego typu przypadkach oraz przykładowe rozwiązanie.

Kroki postępowania w celu narysowania figury bazującej na rekurencji:

1. Zidentyfikuj element (lub pewien schemat) powtarzający się w ramach całej figury na kolejnych jej poziomach.
2. Znajdź jak najprostszy ciąg instrukcji potrzebny do narysowania takiego elementu i umieść go w funkcji.

3. Dodaj w tak napisanej funkcji jej rekurencyjne wywołania w miejscach, gdzie element się „powtarza”. Zwróć uwagę na kierunek i zwrot obiektu rysującego przed i po wywołaniu funkcji. Dodaj warunek ograniczający dalsze rekurencyjne wywołania funkcji.

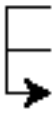
Przykładowo dla prostego fraktala przedstawionego na rys. 1 może to przebiegać następująco:

1. Jako powtarzający się element fraktala przyjmujemy fragment przedstawiony na rys. 4 i 5.
2. Aby narysować taki fragment wystarczy wykonać ciąg instrukcji, które przedstawiono poniżej w funkcji `frac_widly`:

```
1     def frac_widly(x):
2           #rysuj lewa część wideł (rys. 2)
3           t.left(90)
4           t.forward(x)
5           t.right(90)
6           t.forward(x)
7           #cofnij się
8           t.backward(x)
9           t.right(90)
10          t.forward(x)
11          #rysuj środkową część wideł (rys. 3)
12          t.left(90)
13          t.forward(x)
14          #cofnij się
15          t.backward(x)
16          #rysuj prawą część wideł (rys. 4)
17          t.right(90)
18          t.forward(x)
19          t.left(90)
20          t.forward(x)
21          #cofnij się do punktu początkowego
22          #i obróć się w kierunku początkowym (rys. 5)
23          t.backward(x)
24          t.left(90)
25          t.forward(x)
26          t.right(90)
```

Rysunek 2:

Rysunek 3:



Rysunek 4:

Rysunek 5:

3. Dodajemy rekurencyjne wywołania `frac_widly(x/3)` w odpowiednich miejscach w funkcji (gdzie element zaczyna się powtarzać z innym wymiarem). W tym przypadku należy wstawić je między liniami 6 i 7, 13 i 14 oraz 20 i 21. Ponadto na początku definicji funkcji `frac_widly` należy dodać jeszcze warunek, który ograniczy liczbę wywołań rekurencyjnych:

```
if x<=1:
      return
```

Chcąc narysować fraktal z rys. 1 wystarczy już tylko przypisanie zmiennej `x` wartości, narysowanie pierwszego odcinka o długości `x` i wywołanie funkcji `frac_widly` z tym samym parametrem `x`:

```
x = 64
t.forward(x)
frac_widly(x)
```

Inne przydatne uwagi

Pisząc tego typu funkcje rekurencyjne warto zwrócić uwagę na pewne właściwości zaprezentowanych wcześniej metod na obiekcie typu `Turtle`, które mogą być przydatne przy rysowania niektórzy figur:
`backward(x)` to inaczej `forward(-x)`
`left(x)` to inaczej `right(-x)`
`left(360 - x)` to inaczej `right(x)`